#3
Claimed
10-3-01

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application: ２０００年　６月２１日

出 願 番 号
Application Number: 特願２０００－１８６４１０

出 願 人
Applicant (s): 三菱電機株式会社

２０００年　７月１４日

特 許 庁 長 官
Commissioner,
Patent Office 及川耕造

出証番号　出証特２０００－３０５５３２０

【書類名】　特許願
【整理番号】　523818JP01
【提出日】　平成12年 6月21日
【あて先】　特許庁長官殿
【国際特許分類】　G05B 19/00
【発明者】
　【住所又は居所】　東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
　【氏名】　仲井　勘
【発明者】
　【住所又は居所】　東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
　【氏名】　林　鋭志
【特許出願人】
　【識別番号】　000006013
　【住所又は居所】　東京都千代田区丸の内二丁目２番３号
　【氏名又は名称】　三菱電機株式会社
【代理人】
　【識別番号】　100062144
　【弁理士】
　【氏名又は名称】　青山　葆
【選任した代理人】
　【識別番号】　100086405
　【弁理士】
　【氏名又は名称】　河宮　治
【手数料の表示】
　【予納台帳番号】　013262
　【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　デバイス制御プログラム開発手段及び実行手段

【特許請求の範囲】

【請求項１】　１つまたは複数の被制御対象オブジェクトを制御するプログラムを作成するプログラム開発手段と、上記の記述されたプログラムを実行するプログラム実行手段において、

上記プログラム開発手段及びプログラム実行手段にて使用されるソフトウエアにおいて上記被制御対象オブジェクトの各々に固有されるソフトウエアモジュールが独立して存在し、

上記ソフトウエアモジュールは、

・プログラム開発手段を構成する表示領域で表示される１つまたは複数のアイコンと、

・上記被制御対象オブジェクトを制御するための方法を記述するための手続きと、

・上記にて記述された被制御対象オブジェクトの制御方法を上記プログラム実行手段にて実行する手続きと

のうちの、少なくとも１つを有し、

上記被制御対象オブジェクトソフトウエアモジュールはそのグローバルＩＤ若しくは相当のデータにより特定される、

プログラム開発手段及びプログラム実行手段。

【請求項２】　１つまたは複数のデバイスが被制御対象オブジェクトとして接続されている場合には、該デバイスからグローバルＩＤ若しくはそれに相当するデータが取得され、上記デバイスに対応する被制御対象オブジェクトソフトウエアモジュールはそのグローバルＩＤ若しくは相当のデータにより特定される、請求項１に記載のプログラム開発手段及びプログラム実行手段。

【請求項３】　上記デバイスがそれに係る被制御対象オブジェクトソフトウエアモジュールを備えており、

そのデバイスが接続され、

そのデバイスからそれに係る被制御対象オブジェクトソフトウエアモジュールが

取得される、

請求項２に記載のプログラム開発手段及びプログラム実行手段。

【請求項４】　複数の被制御対象オブジェクトソフトウエアモジュールを備えるデータベースサーバから、

通信回線を介して、接続されているデバイス若しくは接続され得るデバイスを含む被制御対象オブジェクトに係る被制御対象オブジェクトソフトウエアモジュールが取得される、

請求項２に記載のプログラム開発手段及びプログラム実行手段。

【請求項５】　上記プログラム開発手段が、接続されているデバイス若しくは接続され得るデバイスを含む被制御対象オブジェクトを、アイコンにより表示する被制御対象オブジェクト表示領域を備える、

請求項２に記載のプログラム開発手段及びプログラム実行手段。

【請求項６】　上記被制御対象オブジェクトソフトウエアモジュールが、被制御対象オブジェクトの現在の状態に対応する複数種類のアイコンを有する、

請求項５に記載のプログラム開発手段及びプログラム実行手段。

【請求項７】　上記プログラム開発手段が、プログラム作成領域を備え、

上記被制御対象オブジェクト表示領域に表示された被制御対象オブジェクトに係るアイコンを、該プログラム作成領域へ追加し、

追加されたそれら被制御対象オブジェクトに係るアイコンによりプログラムを構成する、

請求項５に記載のプログラム開発手段及びプログラム実行手段。

【請求項８】　上記被制御対象オブジェクトソフトウエアモジュールが有する、上記被制御対象オブジェクトを制御するための方法を記述するための手続きを使って、該被制御対象オブジェクトの動作を設定していくことによりプログラムを構成する、

請求項７に記載のプログラム開発手段及びプログラム実行手段。

【請求項９】　上記被制御対象オブジェクトソフトウエアモジュールが、被制御対象オブジェクトの設定動作に対応する複数種類のアイコンを有する、

請求項８に記載のプログラム開発手段及びプログラム実行手段。

【請求項１０】　プログラム作成領域において、被制御対象オブジェクトアイコンを含む複数のアイコンを接続してフローチャートを形成することにより、プログラムの作成を行なう、
請求項７に記載のプログラム開発手段及びプログラム実行手段。

【請求項１１】　上記被制御対象オブジェクト表示領域が、プログラムのシミュレーションを実行する際に、被制御対象オブジェクトの想定される状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、シミュレーション表示領域として利用される、
請求項９に記載のプログラム開発手段及びプログラム実行手段。

【請求項１２】　上記プログラム実行手段が上記被制御対象オブジェクトを制御するプログラムを実行する際に、
上記被制御対象オブジェクト表示領域が、被制御対象オブジェクトの状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、モニタリング表示領域として利用される、
請求項６に記載のプログラム開発手段及びプログラム実行手段。

【請求項１３】　上記プログラム実行手段にて稼動する、上記被制御対象オブジェクト制御プログラムが、グローバルＩＤを備えるデバイスを含む被制御対象オブジェクトに対して、メッセージを送信または受信するプログラム記述形態である、
請求項７に記載のプログラム開発手段及びプログラム実行手段。

【請求項１４】　１つまたは複数のデバイスが被制御対象オブジェクトとして接続されている接続形態は、プラグ・アンド・プレイ機能又は活線抜粋をサポートするインタフェースであることを特徴とする、
請求項２に記載のプログラム開発手段及びプログラム実行手段。

【請求項１５】　請求項７に記載のプログラム開発手段またはプログラム実行手段を構成し稼動させる、プログラムコードを、記録したことを特徴とする記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、自動機アプリケーションシステムを開発し実行するシステムに関する。

【０００２】

【従来の技術】

以下のような自動機一般を構築する際には、例えばＰＬＣ（Ｐｒｏｇｒａｍｍａｂｌｅ　Ｌｏｇｉｃ　Ｃｏｎｔｒｏｌｌｅｒ）やモーションコントローラなどのコントローラが利用される。

・ＦＡ（Ｆａｃｔｏｒｙ　Ａｕｔｏｍａｔｉｏｎ）分野で用いられる工作機械

・産業用ロボットなどの産業用自動機

・自律ロボット

・各種自動設備などの一般用途の自動機

それらコントローラには、コントローラに信号を入力するセンサやスイッチなどの各種入力デバイスが接続される。同様にして、コントローラにより制御されるモータや表示器などの各種出力デバイスが接続される。また、コントローラには、それら入出力デバイスへの指令（指示）を記述するプログラムが搭載されている。プログラムに記述される指令（指示）が入出力デバイスに伝えられることにより、自動機は動作する。

【０００３】

上記の入出力デバイスは、コントローラの入出力コネクタ（即ち、入出力ポート）に接続される。従来のコントローラを利用する場合に、プログラム作成者は、コントローラのどの（何番目の）入出力ポートにどの種類の入出力デバイスを接続するか（したか）を正確に把握する必要があった。その上で、コントローラ上で使用される、デバイスに係るソフトウエアモジュールや動作プログラムの設定が、正確に行われた。そうしなければ、コントローラが該デバイスに適切な指令（指示）を送って正しく制御することができないのである。なお、デバイスに係るソフトウエアモジュールには、例えばデバイスドライバが含まれる。

【０００４】

パソコン（パーソナルコンピュータ）に対し拡張カードや周辺デバイスを追加

する場合にも、数年前までは上記と同様の事情が存在した。即ち、まずパソコン利用者は、装置本体の備える入出力ポートのどのものに、何の種類の周辺デバイスを接続するか（したか）を正確に把握する必要があった。その上で、パソコンが使用する、デバイスに係るソフトウエアモジュールや動作プログラムの設定が、正確に行われた。

【０００５】

ところで、近年、パソコンとその周辺デバイスとの接続に関しては、プラグ・アンド・プレイという概念をもつインタフェースが開発された。そのインタフェースの利用により、パソコン利用者の上記のような負担が大幅に軽減されている。例えば、「ＵＳＢ」や「ＩＥＥＥ１３９４」のようなインタフェースがその例である。これらのインタフェースをサポートする周辺デバイスには、グローバルユニークＩＤ（ＧＵＩＤ）と呼ばれる識別子が保持されている。

【０００６】

ＧＵＩＤは、各種デバイス別に（世界で）唯一割り当てられる識別子である。パソコンが、各種デバイスに各々対応するデバイスドライバなどのソフトウエアモジュールを予め相当量保持している場合を想定する。まず、入出力デバイスが接続されると該デバイスからＧＵＩＤが取得される。パソコンは、そのＧＵＩＤにより適切なソフトウエアモジュールを絞り込んで確定し、そして利用することができる。この機能によって、パソコン利用者が、デバイスドライバなどのソフトウエアモジュールの設定を行なうことの必要がなくなる。即ち、パソコン利用者はパソコン内にそのようなソフトウエアモジュールが存在していることすら意識することがない。にもかかわらず、デバイスを接続してデバイスを使用することが可能となる。但し、当該デバイスを使用する際のパラメータの調節が、パソコン利用者のなすべき作業として残っている。けれども、予め設定されている初期値をそのまま使用する限りこの作業は必要がない。

【０００７】

ＵＳＢやＩＥＥＥ１３９４にはさらにホットプラグ（活線抜粋）の概念の機能も備わる。従来のパソコンでは、新たに接続されたデバイスに関するソフトウエアモジュールの設定を有効にするには、一度パソコンの電源を切り、そして再起

動する必要があった。上記のホットプラグ（活線抜粋）の機能は、パソコンの電源を切ること無しにデバイスを接続して使用することを可能とするものである。このホットプラグは、デバイス接続に関する設定での作業ををほぼ皆無にするものであり、パソコン使用者への負担を従来より大幅に軽減する。

【０００８】

ＵＳＢやＩＥＥＥ１３９４といったプラグ・アンド・プレイの概念を持つインタフェースには、さらに別の特徴が備わる。それは、通信プロトコルが一般に公開されている（即ち、オープン性を有している）ことである。これにより、様々なサードパーティがパソコン周辺機器を作成することが可能である。また、従来、特定のデバイスをパソコンに接続するためには、特定の接続ボードなどの拡張ボードをパソコンに接続しなければならなかった。しかし、上記の特徴によりその煩わしさも無くなった。

【０００９】

このオープン性は、パソコンとその周辺デバイスにおいて実現されているものである。これに留まらずコントローラとその周辺デバイスの間の通信インタフェースにおいても利用されるニーズが高まっている。ＵＳＢやＩＥＥＥ１３９４のような、オープン性を有するインタフェースの機能を備えるコントローラが試行されている。

【００１０】

ところで、上述のように、近年のコントローラ及びパソコンは、確かに、ＵＳＢやＩＥＥＥ１３９４などのインタフェースを採用しつつある。しかしながら、それらのプラグ・アンド・プレイ機能が十分に生かされているとは言い難い状況にある。即ち、ただデバイスを接続するだけで即座にデバイスを使用することが可能となる、という状況は実現されていない。

【００１１】

まず、パソコンとその周辺デバイスについて説明する。例えばパソコンの一般的なマウスの場合、マウスを動かせばモニタ上の矢印が動くように設定されている。つまり、マウスがパソコンとの関係においてどのように動作するのかは予め決められ固定されている。あるいは、ＵＳＢカメラを例に挙げる。ＵＳＢカメラ

からの画像をパソコンディスプレイに表示するための特定のプログラムは、通常、ＵＳＢカメラの製作者からＵＳＢカメラと共に配布されている。しかも、そのプログラムに拠らなければ、そのＵＳＢカメラは動作しない。即ち、パソコンに接続される周辺デバイスは、その用いられ方とパソコンとの関係における動作とが予め決められている。加えて、パソコン利用者がそのデバイスの用いられ方と動作とを自由に変更する（即ち、プログラムコードにより指示する）ことができない。

【００１２】

コントローラと周辺デバイスとについても説明する。そもそもコントローラにおいては、接続するデバイスの用いられ方をプログラムコードにより指示することが不可欠である。即ち、デバイスを接続すれば、即座にデバイスを使用することが可能になる、とはもともとなり得ない。必ずプログラムコードによるプログラミングが要求される。

【００１３】

しかも、コントローラ上で使用されるソフトウエアモジュールや動作プログラムの設定に関しては、従来のコントローラと同様に全て正確に行なう必要がある。ＵＳＢやＩＥＥＥ１３９４といったインタフェースを採用しているとはいえ、それらのプラグ・アンド・プレイ機能の利点が未だ十分には生かされていない。

【００１４】

【発明が解決しようとする課題】

本発明は、プラグ・アンド・プレイ機能を備えるインタフェースを用いるデバイスやコントローラによって自動機を構築する際にて、利用されることを意図する。第１に、当該デバイスやコントローラに係るセットアップの手間を大幅に削減することを目的とする。第２に、シーケンスプログラム、イベントドリブンプログラム、モーションプログラムなどの各種制御プログラム作成を、簡素化することを目的とする。

【００１５】

【課題を解決するための手段】

本発明は、上記の目的を達成するためになされたものである。本発明に係る請

求項１に記載のプログラム開発手段及びプログラム実行手段は、

１つまたは複数の被制御対象オブジェクトを制御するプログラムを作成するプログラム開発手段と、上記の記述されたプログラムを実行するプログラム実行手段である。

上記プログラム開発手段及びプログラム実行手段にて使用されるソフトウエアにおいて上記被制御対象オブジェクトの各々に固有されるソフトウエアモジュールが独立して存在し、

上記ソフトウエアモジュールは、

・プログラム開発手段を構成する表示領域で表示される１つまたは複数のアイコンと、

・上記被制御対象オブジェクトを制御するための方法を記述するための手続きと、

・上記にて記述された被制御対象オブジェクトの制御方法を上記プログラム実行手段にて実行する手続きと

のうちの、少なくとも１つを有し、

上記被制御対象オブジェクトソフトウエアモジュールはそのグローバルＩＤ若しくは相当のデータにより特定される。

【００１６】

本発明に係る請求項２に記載のプログラム開発手段及びプログラム実行手段は、

１つまたは複数のデバイスが被制御対象オブジェクトとして接続されている場合には、該デバイスからグローバルＩＤ若しくはそれに相当するデータが取得され、上記デバイスに対応する被制御対象オブジェクトソフトウエアモジュールはそのグローバルＩＤ若しくは相当のデータにより特定される、

請求項１に記載のプログラム開発手段及びプログラム実行手段である。

【００１７】

本発明に係る請求項３に記載のプログラム開発手段及びプログラム実行手段は、

上記デバイスがそれに係る被制御対象オブジェクトソフトウエアモジュールを

備えており、
そのデバイスが接続され、
そのデバイスからそれに係る被制御対象オブジェクトソフトウエアモジュールが取得される、
請求項２に記載のプログラム開発手段及びプログラム実行手段である。

【００１８】

本発明に係る請求項４に記載のプログラム開発手段及びプログラム実行手段は、

複数の被制御対象オブジェクトソフトウエアモジュールを備えるデータベースサーバから、

通信回線を介して、接続されているデバイス若しくは接続され得るデバイスを含む被制御対象オブジェクトに係る被制御対象オブジェクトソフトウエアモジュールが取得される、
請求項２に記載のプログラム開発手段及びプログラム実行手段である。

【００１９】

本発明に係る請求項５に記載のプログラム開発手段及びプログラム実行手段は、

上記プログラム開発手段が、接続されているデバイス若しくは接続され得るデバイスを含む被制御対象オブジェクトを、アイコンにより表示する被制御対象オブジェクト表示領域を備える、
請求項２に記載のプログラム開発手段及びプログラム実行手段である。

【００２０】

本発明に係る請求項６に記載のプログラム開発手段及びプログラム実行手段は、

上記被制御対象オブジェクトソフトウエアモジュールが、被制御対象オブジェクトの現在の状態に対応する複数種類のアイコンを有する、
請求項５に記載のプログラム開発手段及びプログラム実行手段である。

【００２１】

本発明に係る請求項７に記載のプログラム開発手段及びプログラム実行手段は

、

上記プログラム開発手段が、プログラム作成領域を備え、

上記被制御対象オブジェクト表示領域に表示された被制御対象オブジェクトに係るアイコンを、該プログラム作成領域へ追加し、

追加されたそれら被制御対象オブジェクトに係るアイコンによりプログラムを構成する、

請求項５に記載のプログラム開発手段及びプログラム実行手段である。

【００２２】

本発明に係る請求項８に記載のプログラム開発手段及びプログラム実行手段は

、

上記被制御対象オブジェクトソフトウエアモジュールが有する、上記被制御対象オブジェクトを制御するための方法を記述するための手続きを使って、該被制御対象オブジェクトの動作を設定していくことによりプログラムを構成する、

請求項７に記載のプログラム開発手段及びプログラム実行手段である。

【００２３】

本発明に係る請求項９に記載のプログラム開発手段及びプログラム実行手段は

、

上記被制御対象オブジェクトソフトウエアモジュールが、被制御対象オブジェクトの設定動作に対応する複数種類のアイコンを有する、

請求項８に記載のプログラム開発手段及びプログラム実行手段である。

【００２４】

本発明に係る請求項１０に記載のプログラム開発手段及びプログラム実行手段は、

プログラム作成領域において、被制御対象オブジェクトアイコンを含む複数のアイコンを接続してフローチャートを形成することにより、プログラムの作成を行なう、

請求項７に記載のプログラム開発手段及びプログラム実行手段である。

【００２５】

本発明に係る請求項１１に記載のプログラム開発手段及びプログラム実行手段

は、

上記被制御対象オブジェクト表示領域が、プログラムのシミュレーションを実行する際に、被制御対象オブジェクトの想定される状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、シミュレーション表示領域として利用される、

請求項９に記載のプログラム開発手段及びプログラム実行手段である。

【００２６】

本発明に係る請求項１２に記載のプログラム開発手段及びプログラム実行手段は、

上記プログラム実行手段が上記被制御対象オブジェクトを制御するプログラムを実行する際に、

上記被制御対象オブジェクト表示領域が、被制御対象オブジェクトの状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、モニタリング表示領域として利用される、

請求項６に記載のプログラム開発手段及びプログラム実行手段である。

【００２７】

本発明に係る請求項１３に記載のプログラム開発手段及びプログラム実行手段は、

上記プログラム実行手段にて稼動する、上記被制御対象オブジェクト制御プログラムが、グローバルＩＤを備えるデバイスを含む被制御対象オブジェクトに対して、メッセージを送信または受信するプログラム記述形態である、

請求項７に記載のプログラム開発手段及びプログラム実行手段である。

【００２８】

本発明に係る請求項１４に記載のプログラム開発手段及びプログラム実行手段は、

１つまたは複数のデバイスが被制御対象オブジェクトとして接続されている接続形態は、プラグ・アンド・プレイ機能又は活線抜粋をサポートするインタフェースであることを特徴とする、

請求項２に記載のプログラム開発手段及びプログラム実行手段である。

【００２９】

本発明に係る請求項１５に記載の記録媒体は、

請求項７に記載のプログラム開発手段またはプログラム実行手段を構成し稼動させる、プログラムコードを、記録したことを特徴とする記録媒体である。

【００３０】

【発明の実施の形態】

以下、図面を参照しつつ、本発明に係る好適な実施の形態について説明する。

【００３１】

実施の形態１．

図１は、本発明の実施の形態１に従い、自動機アプリケーションシステムを構築する手順の一例を示したものであり、以下この手順に従って説明を行う。

【００３２】

アプリケーションを構築する際には、まず、

①デバイスをコントローラに接続して自動機を組み立てる（図１・Ｓ０２）。

【００３３】

図２（１）は、コントローラ６と周辺デバイス１０とを接続する一例を示す。プログラム作成手段２とコントローラ６とは、第１の接続手段４により接続される。コントローラ６と周辺デバイス１０とは、第２の接続手段８とにより接続される。プログラム作成手段２は、一般的にはパソコン（パーソナル・コンピュータ）である。オペレーティングシステムとしては、一般的にはＭｉｃｒｏｓｏｆｔ（マイクロソフト）社の「Ｗｉｎｄｏｗｓ９８」、「ＷｉｎｄｏｗｓＮＴ」若しくは「Ｗｉｎｄｏｗｓ２０００」が利用されるが、アップルコンピュータ社の「ＭａｃＯＳ」や、更にはＬｉｎｕｘなどでもよい。またプログラム作成手段２は、テレビジョン受像機と接続して用いるゲーム機や、表示画面を備えて単体で用いるゲーム機（例えば、携帯用ゲーム機）などであってもよい。

【００３４】

また、プログラム作成手段２がコントローラ６と一体であってもよい。図２（２）はその一例であって、コントローラ６がプログラム作成手段２の機能を有する。この例では、コントローラ６は、プログラムを作成する際に使用する表示画

面や設定ボタンを有しているのが好ましい。その表示画面がタッチパネル等の機能を備える場合には設定ボタンは備わらなくてもよい。図２（３）は、プログラム作成手段２がコントローラ６と一体である別の例であって、プログラム作成手段２がコントローラ６の機能を有する。

【００３５】

第１の接続手段４、第２の接続手段８は、同じ接続形態でもそれぞれ異なる接続形態でもよい。また、ＵＳＢやＩＥＥＥ１３９４等の有線でもＢｌｕｅｔｏｏｔｈやＩｒＤＡ等の無線でもよい。

【００３６】

次に、②接続されている周辺デバイス１０の情報を、第１の接続手段４及び第２の接続手段８を介して、コントローラ６から取得する（図１・Ｓ０４）。各デバイス１０は、そのデバイス１０の製作者（製作会社）及び機種を識別するための、ＧＵＩＤ（グローバルユニークＩＤ）と呼ばれる識別子を有する。識別子に相当するデータを有する形態でもよい。それら識別子（または識別子相当のデータ）は、メモリ部（図示せず。）に格納される。プログラム作成手段２上で稼動するプログラム開発環境２０は、ＧＵＩＤを収集する。このことにより、コントローラ６に接続されている周辺デバイス１０が何であるかを把握することができる。

【００３７】

次に、③接続されている周辺デバイス１０に対応するデバイスドライバをリンク（関連付け）する（図１・Ｓ０６）。プログラム開発環境２０は、利用し得る全てのデバイスに対応するデバイスドライバ群を有したデータベースにアクセスする。そして、ＧＵＩＤに対応するデバイス１０のデバイスドライバに対してリンクを張る。または、当該デバイスドライバをプログラム開発環境２０内に持ってくる（即ち、例えば、コピーする、若しくはダウンロードする）。デバイスドライバはＤＬＬ（ダイナミック・リンク・ライブラリ）の形態で提供されても構わない。

【００３８】

図３は、本発明の実施の形態１に係るソフトウエアアーキテクチャの一例を示

した図である。該アーキテクチャには、プログラム開発環境２０、プログラム実行環境２２、及び、デバイス１０に対応するデバイスドライバ２４が含まれる。図３は、コントローラ６が、プログラム作成手段２の機能を有する場合を示すものである。コントローラ６とプログラム作成手段２とが、別々に存在する場合も、もちろん想定され得る。この場合、プログラム作成手段２がプログラム開発環境２０を有し、コントローラ６がプログラム実行環境２２を有する。プログラム作成手段２は、プログラムを開発する際にシミュレーションを行なうため、プログラム実行環境２２も同時に有しても構わない。

【００３９】

デバイスドライバ２４は、以下のモジュールのうち少なくともいずれか一つを有している。

・プログラム開発環境２０で表示されるアイコン
・当該デバイスを制御するための方法を記述するための手続き
・上記の当該デバイスの制御方法をプログラム実行環境２２上で実行する手続き
・当該デバイスと通信するための通信手続き

図４（１）は、異なるデバイス１０を接続する場合に想定されるソフトウエアアーキテクチャである。

【００４０】

プログラム実行環境２２が、通信手続き２６を提供する場合もある。異なるデバイス１０でも同じ通信手段を使用している場合には、共通の通信手続き２６を利用することになる。図４（２）はその様子を示す。同じ通信手段とは、例えばどちらもＵＳＢデバイスであるような場合である。一方、異なるデバイス１０で異なる通信手段を使用している場合には、異なる通信手続き２６を利用することになる。図４（３）はその様子を示す。異なる通信手段とは、例えば一方がＵＳＢデバイスで他方がＩＥＥＥ１３９４デバイスであるような場合である。

【００４１】

利用し得る全てのデバイス１０に対応するデバイスドライバ群を有したデータベースは、プログラム開発環境２０内に存在してもよい。図５（１）はその例を示し、データベースはパソコン２内のハードディスク４０上に存在する。また、

プログラム開発環境２０外の外部記憶媒体４２に存在してもよい。図５（２）はその例を示し、データベースは、パソコン２でアクセス可能な記憶媒体（ＣＤ－ＲＯＭ、ＤＶＤ－ＲＯＭ、ゲームソフト記憶手段、等）上に存在する。さらに、遠隔地のサーバ内（例えば、図５（３）のようにサーバのデータベース記憶領域４４）に存在してもよい。この場合には図５（３）に示されるように、ＬＡＮやインターネットなどの伝送経路４６を通じてアクセスすることになる。

【００４２】

各周辺デバイス１０が、自らのデバイスドライバ２４を提供するようにしてもよい。即ち、まず、例えば各デバイス１０が、そのデバイスが何であるかを識別するためのＧＵＩＤのみならずデバイスドライバ２４も備えている。そこで、プログラム開発環境２０は接続されているデバイス１０から直接デバイスドライバ２４を取得してもよい。この場合、デバイスドライバ群を有したデータベースにアクセスする必要はない。但し、デバイス１０における記憶領域では、ＧＵＩＤを記憶する領域容量に加えてデバイスドライバ２４を記憶する分だけ大きい領域容量が必要となる。

【００４３】

次に、④プログラム開発環境２０のデバイス表示ウインドウ（領域）６６上にデバイスアイコン６８を表示する（図１・Ｓ０８）。アイコンを有しているデバイスドライバ２４に関しては、当該アイコンを表示する。

【００４４】

図６は、本発明に係るプログラム開発環境２０の一例を示した図である。該プログラム開発環境２０は、以下の要素を含んでいる。

・プログラム開発環境画面全体６０

・プログラム開発環境２０の基本的な操作を取り扱うツールバー６２

・プログラム作成領域６４

・デバイス表示領域６６

プログラム作成領域６４内に、アイコンをドラッグアンドドロップしたり、アイコン同士を接続したりすることで、プログラムを作成することを想定している。デバイス表示領域６６は、構築しようとする自動機アプリケーションシステムで

使用するデバイス１０をアイコン形式で表示する領域である。該領域６６の形状は、図６（１）のように、ウインドウ形式でもよく、図６（２）のように、ツールバーの形式でもよく、更に図６（３）のように、ツリービュー形式でもよい。

【００４５】

次に、⑤デバイスアイコン６８を、デバイス表示領域６６から、プログラム表示領域６４へドラッグ・アンド・ドロップする。即ち、デバイス表示領域６６に表示されているデバイスアイコン６８を、プログラム作成領域６４上に新たに作成する（図１・Ｓ１０）。プログラム開発環境２０は、プログラム作成に必要な種々の制御コマンド（分岐コマンド、繰り返しコマンド、若しくは割込みコマンド等）をアイコン形式で表示する制御コマンド表示領域（図示せず。）を備えているのが好ましい。必要な制御コマンドのアイコンも、プログラム作成領域６４へ、ドラッグアンドドロップする（新たに作成する）。

【００４６】

次に、⑥プログラム作成領域６４上のアイコン同士を接続し、コントローラ６が実行する制御プログラムを作成する（図１・Ｓ１２）。アイコン同士の接続においては、アイコンとアイコンを隣り合わせに接続する方法によってもよい。アイコン同士を線で接続するフローチャート形式による方法でもよい。

【００４７】

図７（１）は、本発明に係るプログラム開発の一例を示した図である。図７（１）におけるプログラム開発環境画面６０では、以下の要素が含まれる。

・デバイス表示領域６６からプログラム作成領域６４へドラッグ・アンド・ドロップされたデバイスアイコン６８’

・プログラムの開始を示している制御コマンドアイコンである開始アイコン７０

・開始アイコン７０とデバイスアイコン６８’とをフローチャート形式に則り接続する矢印７２

・デバイスアイコン６８’の動作を記述するためのダイアログボックス（またはアプリケーションソフトウエア）８８

【００４８】

ダイアログボックス８８は、デバイスアイコン６８’がダブルクリックされる

ことにより開かれる。デバイスドライバ２４が該デバイス１０を制御するための方法を記述するための手続きを備える場合は、その手続きに従ったダイアログボックスが提示される。デバイス表示領域６６からプログラム作成領域６４へドラッグ・アンド・ドロップされた複数のデバイスアイコン６８’をプログラミングしてから（即ち、デバイスアイコン６８’同士を接続してから）、各アイコン６８’に対してダイアログボックス８８を開いて動作記述をしてもよい。各アイコン６８’に対してダイアログボックス８８を開いて動作記述をしてから、アイコン６８’同士を接続してもよい。順序はどのようでも構わない。デバイスドライバ２４が、デバイス１０の設定動作や現在の状況に応じた複数種類のアイコンを有している場合は、その動作設定に従ったアイコンが表示される（例えば、図７（２）のアイコン６８”）。

【００４９】

上記のデバイスアイコン６８’の動作の記述において、アイコン動作設定用のアプリケーションソフトウエアを立ち上げて動作を設定する、というように構成することも可能である。この場合、より複雑な動作を記述することができる。

【００５０】

最後に、⑦作成されたプログラムとデバイスドライバ２４の必要部分とを、プログラム実行環境２２（コントローラ６）へ転送し、そして実行する（図１・Ｓ１４）。例えば、プログラム実行環境２２が通信手続きを提供する場合には、通信手続きに関連する部分は転送される必要はない。従って、デバイスドライバ２４に関しては、必要部分のみ転送される。

【００５１】

また、コントローラ６がプログラムの進行状況を表示するための表示手段を備えていることがある。このとき、アイコン６８も転送すれば、デバイスの状態に応じたアイコン６８、６８’、６８”をコントローラ６の表示手段に表示することができる。

【００５２】

以上の実施の形態１に係るプログラム開発環境２０及びプログラム実行環境２２の動作は、コンピュータ制御に適うプログラムコードにより記述されるもので

ある。これらプログラムコードが、パソコン（等）のメモリ部（図示せず。）や処理部（図示せず。）にて、格納され稼動制御される。これらプログラムは、ＣＤ－ＲＯＭなどの記録媒体に記録することができる。

【００５３】

実施の形態１を利用することにより、以下のような効果が得られる。

【００５４】

接続されているデバイス１０に対応すべきデバイスドライバ２４を自動的にリンクする。従って、制御プログラム作成においては、当該デバイス１０の動作に関して必要最小限の記述をするだけでよい。即ち、当該デバイス１０に係るその他の設定を一切する必要がない。よって、プラグ・アンド・プレイ機能が十分に生かされ制御プログラムの開発効率が格段に向上する。

【００５５】

プログラム可能なデバイス１０のみが利用可能であるからプログラム作成のミスを未然に防げることが、挙げられる。当該デバイス１０に係るソフトウエアの大半は、デバイスドライバ２４内部にカプセル化されている。従って、それら部分のコーディング内容に関する知識が無くとも動作実証済みのデバイスドライバ２４にリンクすることで大規模なアプリケーションを構築し得る。該構築者が理解する必要があるのは、デバイスドライバ２４によって提供される機能と、該デバイスドライバ２４が使用する外部パラメータのみである。制御プログラム生成者は、外部パラメータの値しか変更できない。よって、整合性のない矛盾したデータ型のデータや制御信号を利用する、というような危険性は大幅に減少するのである。

【００５６】

また、アイコンのドラッグ・アンド・ドロップと、アイコン同士の接続という基礎的な操作のみでプログラム作成が可能である。プログラムは、フローチャート形式に則り作成される。プログラム作成が容易であるだけでなく、作成されたプログラムの可読性もよい。デバイスが、設定や状況に応じて複数種類のアイコンを表示することも、作成されたプログラムの可読性がよいことに貢献する。

【００５７】

実施の形態２．

図８は、本発明の実施の形態２に従い、仮想的に自動機アプリケーションシステムを構築する手順の一例を示したものである。以下、この手順に従って説明を行なう。

【００５８】

仮想アプリケーションを構築する際には、まず、

①自動機をデバイス表示領域６６上で組みたてる。即ち、デバイスアイコン６８を追加する（図８・Ｓ１０２）。

【００５９】

図９は、本発明に係るプログラム開発環境２０の一例を示した図である。該プログラム開発環境２０は、以下の要素を含んでいる。

・プログラム開発環境画面全体６０

・プログラム開発環境２０の基本的な操作を取り扱うツールバー６２

・プログラム作成領域６４

・デバイス表示領域６６

このプログラム開発環境２０は、前に説明した図６（１）と同じであっても構わない。プログラム開発環境２０は、利用し得る全てのデバイス１０を（図９（１）ではダイアログボックス１００によって）提示する。その中から所望のデバイス１０を選択し、デバイスアイコン６８をデバイス表示領域６６に追加する。利用し得る全てのデバイス１０に対応するデバイスドライバ群を有したデータベースは、前に説明した図５のものと同じであってよい。

【００６０】

仮想的に自動機アプリケーションシステムを構築するに際しては、このようにデバイスアイコン６８をデバイス表示領域６６に追加するだけでもよい。どのような自動機アプリケーションシステムを構築するのかを分かりやすくするために、「絵」を書き加えてもよい。図９（２）はその一例で、符号１０２により示されるのが、書き加えられた絵である。

【００６１】

②デバイス１０に対応するデバイスドライバ２４は、デバイスアイコン６８を

デバイス表示領域６６へ追加する際に自動的にリンクされる（図８・Ｓ１０４）。実施の形態１では、ＧＵＩＤを取得してコントローラ６に接続されているデバイス１０が何であるかを判断する。一方、実施の形態２では、利用し得る全てのデバイス１０を画面６０にて提示し、その中から所望のデバイス１０を選択する。よって、その選択により、実施の形態１と同様に、当該デバイスドライバ２４のリンクを行うことができる。

【００６２】

次に、③動作させるアイコン６８を、プログラム作成ウインドウ（領域）６４へドラッグ・アンド・ドロップする（図８・Ｓ１０６）。続いて、

④プログラム作成領域６４上のアイコン同士を接続し、コントローラ６が実行する制御プログラムを作成する（図８・Ｓ１０８）。

このようなプログラムの作成も、実施の形態１と同じである。

【００６３】

⑤シミュレーションにより作成されたプログラムの動作を確認する（図８・Ｓ１１０）。図１０（１）のプログラム作成領域６４は、作成されたプログラムの例を示す。作成したプログラムの動作を確認するためのシミュレーションを、プログラム開発環境２０上で実行することができる。シミュレーションを実行すると、プログラム作成領域６４では、動作プログラム通りに実行していく。その際、プログラム上で現在どのブロック（即ち、アイコン、コマンド）を実行しているのかを表示する。一方デバイス表示領域６６上のデバイスアイコン６８は、そのデバイス１０のシミュレーション時の現況を示している。デバイスドライバ２４が、デバイス１０の設定動作や現在の状況に応じた複数種類のアイコンを有している場合は、その動作設定に従ったアイコンが表示される。図１０（２）はその１例である。プログラム上ではＯＮの所を実行しており、デバイス表示領域６６にあるデバイスアイコン６８は、「ＯＮ」された状態を示している。

【００６４】

ここで作業を終了し、作成したプログラムを保存することもできる。

【００６５】

また、⑥作成されたプログラムとデバイス１０のソフトウエアモジュールをと

コントローラ６へ転送し実行することもできる（図８・Ｓ１１２）。

【００６６】

この場合、以下の２者は同一でなければならない。

（１）プログラム開発環境２０上で仮想的に構築したアプリケーションの構成

（２）実際のコントローラ６とデバイス１０とを接続して実際に構築したアプリケーションの構成

但し、これを確認することは容易である。即ち、作成されたプログラムとデバイス１０のソフトウエアモジュールをコントローラ６へ転送する前に、接続されるデバイス１０のＧＵＩＤを収集すればよい。そうすると、仮想的に構築したアプリケーションのデバイス情報と、実際のＧＵＩＤにより得られる情報とを、容易に比較することができる。もし両者が異なるならば、その旨を示す警告画面をユーザに示す等の処置をとることができる。

【００６７】

たとえ、上記確認作業を行なわずに、作成されたプログラムとデバイス１０のソフトウエアモジュールとをコントローラ６へ転送し実行しても、問題は生じない。コントローラ６上の制御プログラムは、動作させようとするデバイス１０と実際に接続されるデバイス１０とが、一致する場合にのみ機能し得る。所望のデバイスが実際に接続されていない場合には、制御プログラムによる動作に係る指令（指示）は無視される。よって、予期せぬ事態が起こるようなことはなく、フェールセーフである。

【００６８】

従って、制御プログラムに記述されていないデバイス１０が実際にコントローラにつながっていたとしても、同様にフェールセーフとなり問題は起こらない。

【００６９】

シミュレーションで行なった内容は、実際のコントローラ６の稼動状況のモニタリングにも使用することができる。即ち、シミュレーション画面はそのままモニタリング画面として、利用することができる。

【００７０】

以上の実施の形態２に係るプログラム開発環境２０及びプログラム実行環境２

２の動作は、コンピュータ制御に適うプログラムコードにより記述されるものである。これらプログラムコードが、パソコン（等）のメモリ部（図示せず。）や処理部（図示せず。）にて、格納され稼動制御される。これらプログラムは、ＣＤ－ＲＯＭなどの記録媒体に記録することができる。

【００７１】

実施の形態１では、実際に自動機アプリケーションシステムを構築する例を示した。一方。実施の形態２では、仮想的に自動機アプリケーションシステムを構築する例を示した。両者を混合する形態も、実施可能である。

【００７２】

実施の形態２を利用することにより、以下のような効果が得られる。

【００７３】

実際にデバイス１０をコントローラ６やプログラム作成手段２に接続しなくとも（即ち、実際に手元にデバイス１０がなくとも）、そのデバイス１０に係るプログラムを作成することができる。仮想的なアプリケーション構築やプログラム作成においても、接続されるはずのデバイス１０に対応するデバイスドライバ２４は自動的にリンクされる。従って、当該デバイス１０の動作に関して必要最小限の記述をするだけでよい。即ち、当該デバイス１０に係るその他の設定を一切する必要がない。よって、仮想的にプラグ・アンド・プレイ機能が十分に生かされ制御プログラムの開発効率が格段に向上する。

【００７４】

さらには、シミュレーションによりプログラム作成のミスを未然に防ぐことができる。プログラムはフローチャート形式で記述され、且つ現在のシミュレーション状況がフローチャートに従い逐次表示されるので、シミュレーションを理解しやすい。上述のように、デバイスドライバ２４が、デバイス１０の設定動作や現在の状況に応じた複数種類のアイコンを有していることがある。その場合には、その動作設定に従ったアイコンが表示され得るため、仮想的に構築したアプリケーションの動作の状況が分かりやすく表示される。また、上述のように、どのような自動機アプリケーションシステムを構築するのかを分かりやすくするために絵を書き加えることができる。このことにより、仮想的に構築したアプリケー

ションシステムの動作の状況がさらにに分かりやすく表示され得る。

【００７５】

シミュレーションで行なった内容は、実際のコントローラ６の稼動状況のモニタリングにおいても、使用することができる。即ちシミュレーション画面はそのままモニタリング画面として、利用することができる。制御プログラムの開発におけるモニタリング画面構築の作業を省くことができる。

【００７６】

本発明では、実際に手元にデバイス１０が無くとも当該デバイス１０に係わる動作プログラムを記述することができる。この発明を利用すれば、大規模アプリケーションの動作プログラムの開発を行なう場合に、動作プログラムを分割して作成してゆくことが可能である。

【００７７】

実施の形態３．

本発明の実施の形態３は、被制御対象オブジェクトがデバイス１０として実在せずに、モニタリング画面に所謂「バーチャルアシスタント（Ｖｉｒｔｕａｌ Ａｓｓｉｓｔａｎｔ）」の形態として存在するような場合の、一つの例である。ここで、「被制御対象オブジェクト」とは、自動機アプリケーションシステムが制御の対象として認識するオブジェクトである。実存在であるデバイス１０や仮想的存在であるバーチャルアシスタントが、この「被制御対象オブジェクト」に含まれる。バーチャルアシスタント（以下では、バーチャル管理人と称する。）７４は、モニタリング画面上にのみ存在する。

【００７８】

本発明の実施の形態３に従い、仮想的に自動機アプリケーションシステムを構築する手順は、図８の実施の形態２に係る手順と同じでよい。

【００７９】

図１１は、本発明に係るプログラム開発環境２０の一例を示した図である。

【００８０】

自動機アプリケーションシステムを構築するに際しては、モニタリング画面の構築もまた重要な位置を占める。よって、モニタリング画面においては、自動機

アプリケーションシステムで実際に使用されるデバイス１０の現況をモニタリングすることのみに留まらないことが好ましい。即ち、操作者にモニタリング現況の内容をよりわかりやすく示すために、上記の「バーチャル管理人」７４をモニタリング画面に登場させることが好ましい。

【００８１】

実施の形態２の説明で述べたように、シミュレーション画面はそのままモニタリング画面として利用することができる。実施の形態２のプログラム開発環境画面６０では、自動機アプリケーションシステムで実際に使用されるデバイス（被制御対象オブジェクト）１０（に係るデバイスアイコン６８）を、デバイス表示領域６０及びプログラム作成領域６４に配置する。シミュレーション時及びモニタリング時には、そのプログラム作成領域６４及びデバイス表示領域６４にてデバイス１０の動作がデバイスアイコン６８’、６８により示される。以上と同様にして、バーチャル管理人７４をプログラム開発環境画面６０のデバイス表示領域６６及びプログラム作成領域６４にて配置する。そうすれば、モニタリング画面でのみ動作するバーチャル管理人７４’を作成し利用することができる。

【００８２】

図１２は、バーチャル管理人７４に関連するプログラムの開発の一つの例を示す図である。図１２において、条件分岐７６のブロックが利用されている。ダイアログボックス７８により、被制御対象オブジェクト（図１２では、バーチャル管理人）７４’の動作の記述を行なう。ダイアログボックス７８は、例えば、被制御対象オブジェクト（バーチャル管理人）７４’がダブルクリックされることにより開かれる。実施の形態１での、デバイスアイコン６８’の動作を記述するダイアログボックス８８は、所謂「チェック形式」により動作が設定される。一方、図１２（１）では、バーチャルアシスタント７４’の動作を記述するダイアログボックス７８は、所謂「リスト形式」により動作が設定されている。勿論、どちらの形式による動作設定であってもよく、また別の形式であってもよい。

【００８３】

実施の形態１と同じように、バーチャル管理人７８’の動作記述において、アイコン動作設定用のアプリケーションソフトウエアを立ち上げて動作を設定する

、というように構成することも可能である。この場合、より複雑な動作を記述することができる。

【００８４】

作成されたプログラムは、コントローラ６へ転送される。コントローラ６が表示手段を備えない場合（例えば、図２（２））、バーチャル管理人７４に係る部分は、例えば、無視されることになる。このとき、コントローラ６にはプログラム全てが転送され、実行時にコントローラ６が無視するという形態でもよい。また、プログラムを転送する時点で、バーチャル管理人７４に係るプログラムの部分が除かれて、コントローラ６に送られてもよい。

【００８５】

バーチャル管理人（バーチャルアシスタント）７４に対応する実在のデバイス１０は存在しない。但し、実在するデバイス１０が有するグローバルユニークＩＤに相当するＩＤコードを、バーチャル管理人（バーチャルアシスタント）７４に備えさせることが可能である。このとき、例えば、現実のデバイス１０に割り当てられているＧＵＩＤとは全く異なるコードを、上記の「相当するＩＤコード」として設定すればよい。すると、コントローラ６にどんなデバイス１０が接続されても、それらのデバイス１０に対し、バーチャル管理人（バーチャルアシスタント）７４に関連する動作の指示が与えられることは決して無い。よって、予期せぬ事態が起こるようなことは無く、フェールセーフである。

【００８６】

図１２（２）は、作成されたプログラムを実行したときの一例である。この例では、開始後、条件分岐７６により、“ＯＮ”を実行するフローへは進まないで、「バーチャル管理人７４が右手を挙げる」フローに進んでいる、という状況を示している。即ち、図示されるように、デバイス表示領域６６上のバーチャル管理人７４は、右手を挙げている。

【００８７】

以上の実施の形態３に係るプログラム開発環境２０及びプログラム実行環境２２の動作は、コンピュータ制御に適うプログラムコードにより記述されるものである。これらプログラムコードが、パソコン（等）のメモリ部（図示せず。）や

処理部（図示せず。）にて、格納され稼動制御される。これらプログラムは、ＣＤ－ＲＯＭなどの記録媒体に記録することができる。

【００８８】

実施の形態３を利用することにより、以下のような効果が得られる。

【００８９】

モニタリング画面にてより多くの情報を提示することが可能になる。しかもバーチャルアシスタント７４は独自の形状（例えば「人」形状）を有するアイコン（等）により表現されているので、前記情報が分かりやすい形で操作者に与えられる。

【００９０】

以下には、実施の形態１、実施の形態２及び実施の形態３に共通の効果を示す。

【００９１】

ＩＥＥＥ１３９４は、所謂「ホームネットワーク」において利用し得るインタフェースである。従って、本発明の実施の形態にてＩＥＥＥ１３９４を利用すれば、デバイス１０として冷蔵庫、電子レンジ、エアコン、若しくは照明器具などの家電製品を接続することができる。このときコントローラ６には、タイマ動作などをプログラミングしておくことができる。

【００９２】

また、本発明の利用により、特にファクトリオートメーションなどの自動機の分野において、ホットプラグ機能を充分に活用することが可能になる。例えば、ある自動機に既にプログラムされ実行されているアプリケーションシステムに対し、デバイス１０を追加若しくは削除する（引き抜く）場合を想定する。この場合従来であれば、コントローラ６の電源を一旦ＯＦＦにして、プログラムを追加若しくは削除等の変更をする必要があった。ホットプラグ機能を利用して本発明を実施することにより、電源ＯＮのままプログラムの追加や削除等の変更をし、かつデバイス１０を追加若しくは削除する（引き抜く）ことができる。よって、作業負担は大幅に軽減される。

【００９３】

【発明の効果】

本発明に係る請求項１に記載のプログラム開発手段及びプログラム実行手段を利用することにより、使用すべきデバイスドライバ（被制御対象オブジェクトソフトウエアモジュール）の内容を明確なものにすることができる。

【００９４】

本発明に係る請求項２に記載のプログラム開発手段及びプログラム実行手段を利用することにより、使用すべきデバイスドライバ（被制御対象オブジェクトソフトウエアモジュール）の内容を明確なものにすることができる。

【００９５】

本発明に係る請求項３に記載のプログラム開発手段及びプログラム実行手段を利用することにより、適切なデバイスドライバ（被制御対象オブジェクトソフトウエアモジュール）を正しく使用できる。また、プログラム開発手段及びプログラム実行手段において、デバイスドライバ（被制御対象オブジェクトソフトウエアモジュール）群を格納するメモリ領域を多く備える必要がない。

【００９６】

本発明に係る請求項４に記載のプログラム開発手段及びプログラム実行手段を利用することにより、適切なデバイスドライバ（被制御対象オブジェクトソフトウエアモジュール）を正しく使用できる。また、デバイスドライバ（被制御対象オブジェクトソフトウエアモジュール）を一元的に管理できる。さらに、プログラム開発手段及びプログラム実行手段において、デバイスドライバ（被制御対象オブジェクトソフトウエアモジュール）群を格納するメモリ領域を多く備える必要がない。

【００９７】

本発明に係る請求項５に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率が、向上する。

【００９８】

本発明に係る請求項６に記載のプログラム開発手段及びプログラム実行手段を利用することにより、デバイスを含む被制御対象オブジェクトの現況が的確に把

握できる。

【００９９】

本発明に係る請求項７に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率がより向上する。

【０１００】

本発明に係る請求項８に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、アイコンに対し様々な動作を設定できる。

【０１０１】

本発明に係る請求項９に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率がより向上する。

【０１０２】

本発明に係る請求項１０に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率が更に向上する。

【０１０３】

本発明に係る請求項１１に記載のプログラム開発手段及びプログラム実行手段を利用することにより、プログラムのシミュレーションを、目視によりリアルタイムで確認できる。

【０１０４】

本発明に係る請求項１２に記載のプログラム開発手段及びプログラム実行手段を利用することにより、プログラムのモニタリングを、目視によりリアルタイムで確認できる。

【０１０５】

本発明に係る請求項１３に記載のプログラム開発手段及びプログラム実行手段を利用することにより、オブジェクト指向システム開発の考え方に従ってシステム開発を行うことができる。よって開発作業においてプログラムの理解効率や作

成効率を更に向上させることができる。

【０１０６】

本発明に係る請求項１４に記載のプログラム開発手段及びプログラム実行手段を利用することにより、プラグ・アンド・プレイ機能や活線抜粋の機能を充分に利用する自動機アプリケーションシステムの構築が可能になる。

【０１０７】

本発明に係る請求項１５に記載の記録媒体を利用することにより、自動機アプリケーションシステムの開発及び実行を適切に且つ効率よく行うことができる。

【図面の簡単な説明】

【図１】　本発明の実施の形態１に従い自動機アプリケーションシステムを構築する手順の一例を示す図である。

【図２】　プログラム作成手段、コントローラ、及び周辺デバイスの接続の例を示す、ブロック図である。

【図３】　本発明の実施の形態１に係る、ソフトウエアアーキテクチャの例を示すブロック図である。

【図４】　複数のデバイスが接続される、本発明の実施の形態１に係るソフトウエアアーキテクチャの例を示すブロック図である。

【図５】　ソフトウエアモジュールデータベースの接続の形態を示す模式図である。

【図６】　プログラム開発環境の模式図である。

【図７】　プログラム開発例の図である。

【図８】　本発明の実施の形態２に従い仮想的に自動機アプリケーションシステムを構築する手順の一例を示す図である。

【図９】　仮想プログラム開発環境の模式図である。

【図１０】　プログラム開発例とシミュレーション例の図である。

【図１１】　バーチャルアシスタントに係るプログラム開発環境の模式図である。

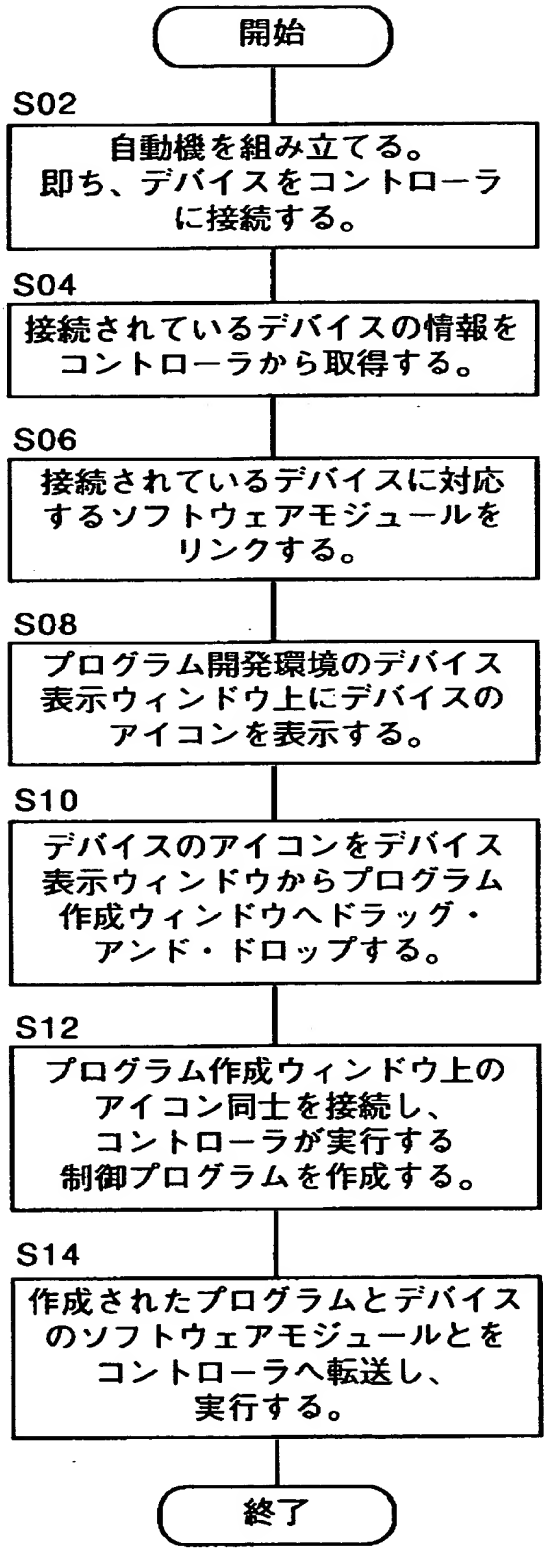
【図１２】　プログラム開発例とモニタリング例の図である。

【符号の説明】

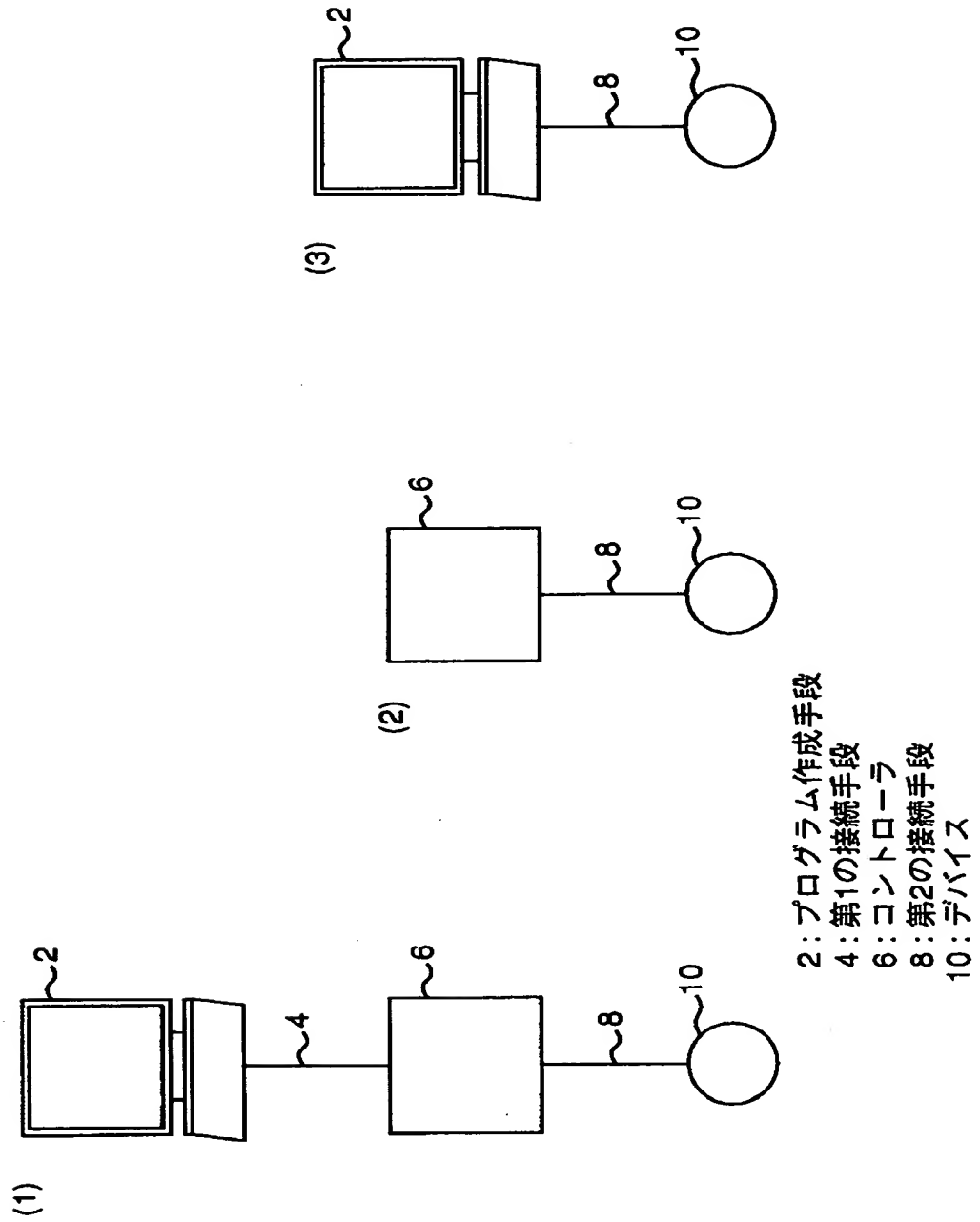
２　プログラム作成手段、４　第１の接続手段、６　コントローラ、８　第２の接続手段、１０　デバイス、２０　プログラム開発環境、２２　プログラム実行環境、２４　デバイスドライバ、２６　通信手続き、４０　ハードディスク、４２　外部記憶媒体、４４　サーバ・データベース記憶領域、４６　伝送手段、６０　プログラム開発環境画面、６２　ツールバー、６４　プログラム作成領域、６６　デバイス表示領域、６８、６８’、６８”　デバイスアイコン、７２　矢印、７４　バーチャル管理人、７６　条件分岐ブロック、７８　ダイアログボックス（またはアプリケーションソフトウエア）、１００　デバイスダイアログボックス、１０２　絵。

【書類名】　図面

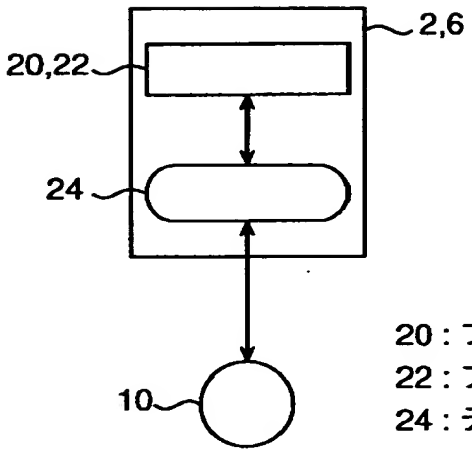
【図１】


開始
S02
自動機を組み立てる。
即ち、デバイスをコントローラ
に接続する。
S04
接続されているデバイスの情報を
コントローラから取得する。
S06
接続されているデバイスに対応
するソフトウェアモジュールを
リンクする。
S08
プログラム開発環境のデバイス
表示ウィンドウ上にデバイスの
アイコンを表示する。
S10
デバイスのアイコンをデバイス
表示ウィンドウからプログラム
作成ウィンドウへドラッグ・
アンド・ドロップする。
S12
プログラム作成ウィンドウ上の
アイコン同士を接続し、
コントローラが実行する
制御プログラムを作成する。
S14
作成されたプログラムとデバイス
のソフトウェアモジュールとを
コントローラへ転送し、
実行する。
終了

【図２】

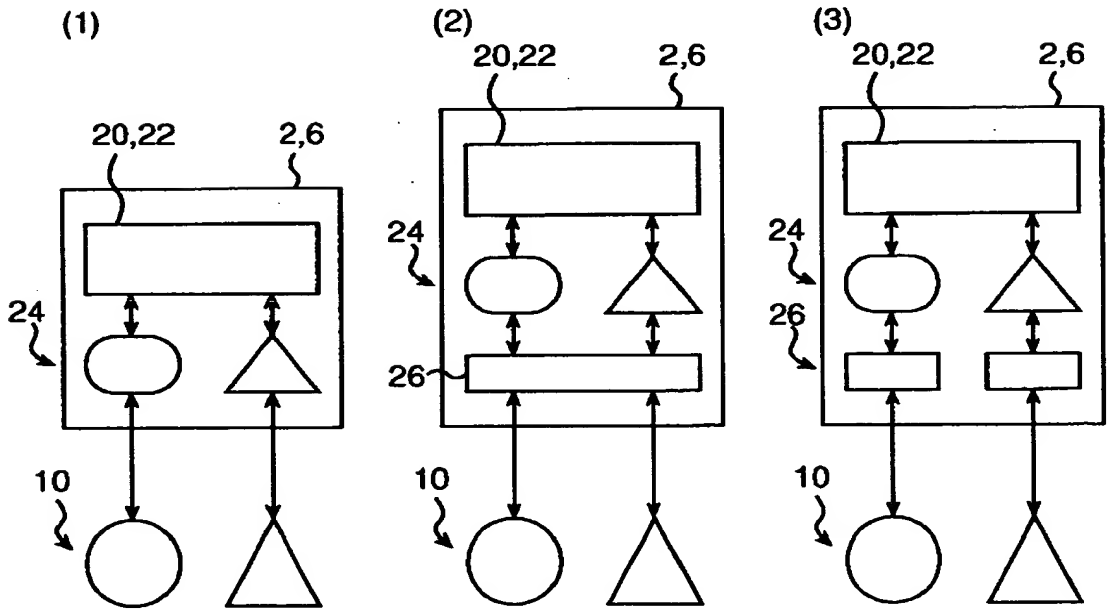

2
10
8
(3)
6
8
10
(2)
2
4
6
8
10
(1)
2：プログラム作成手段
4：第1の接続手段
6：コントローラ
8：第2の接続手段
10：デバイス

【図３】



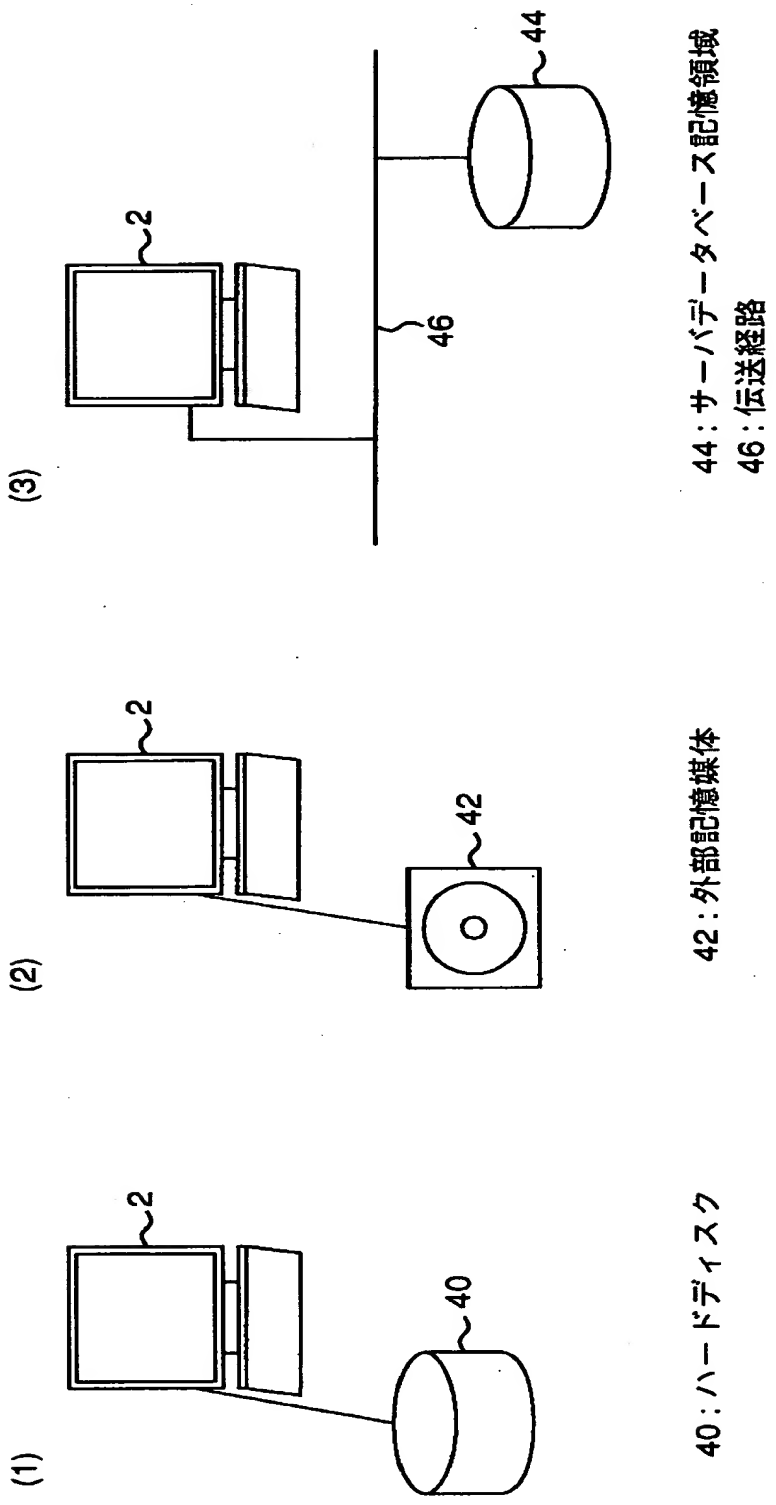
20：プログラム開発環境
22：プログラム実行環境
24：デバイスドライバ

【図４】

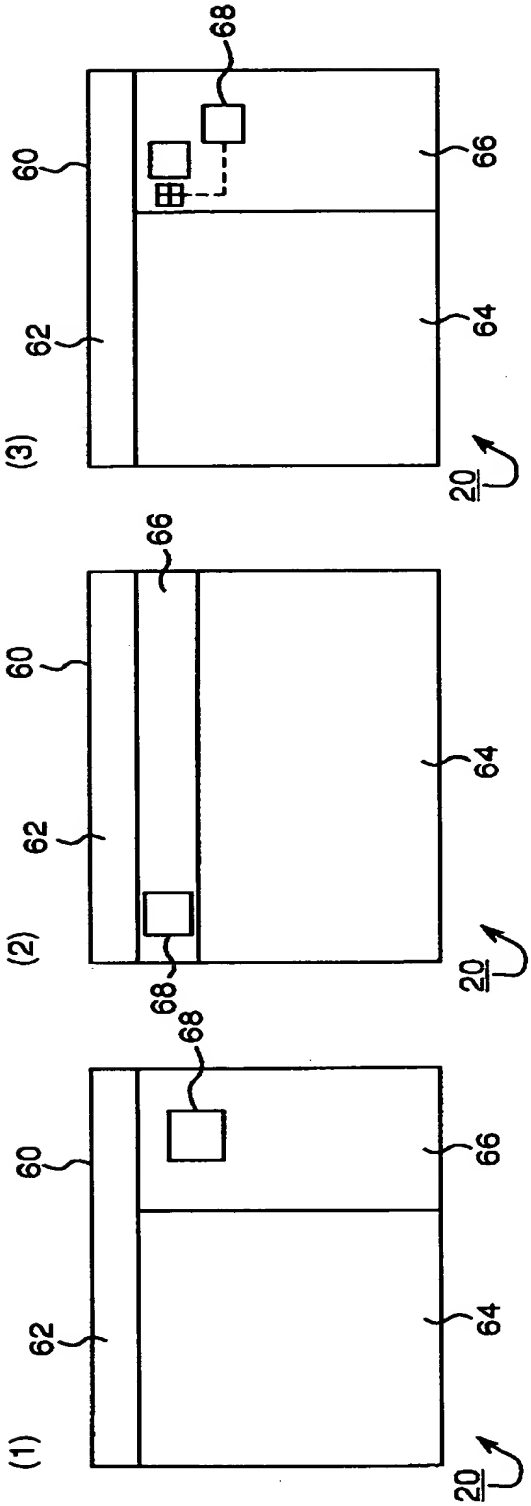


26：通信手続き

【図５】

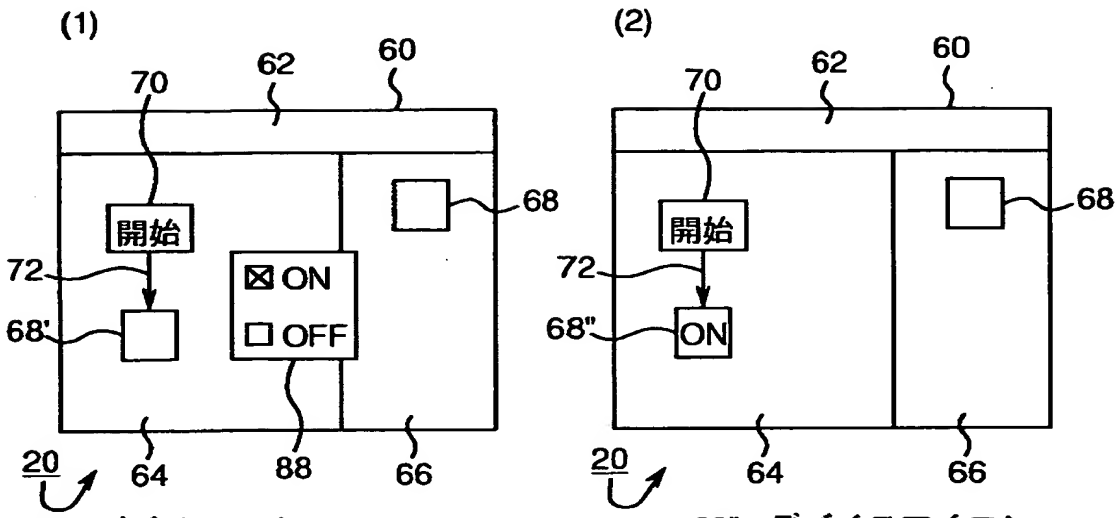

(1)
2
40
(2)
2
42
(3)
2
44
46
40：ハードディスク
42：外部記憶媒体
44：サーバデータベース記憶領域
46：伝送経路

【図６】



60：プログラム開発環境画面
62：ツールバー
64：プログラム作成領域
66：デバイス表示領域
68：デバイスアイコン

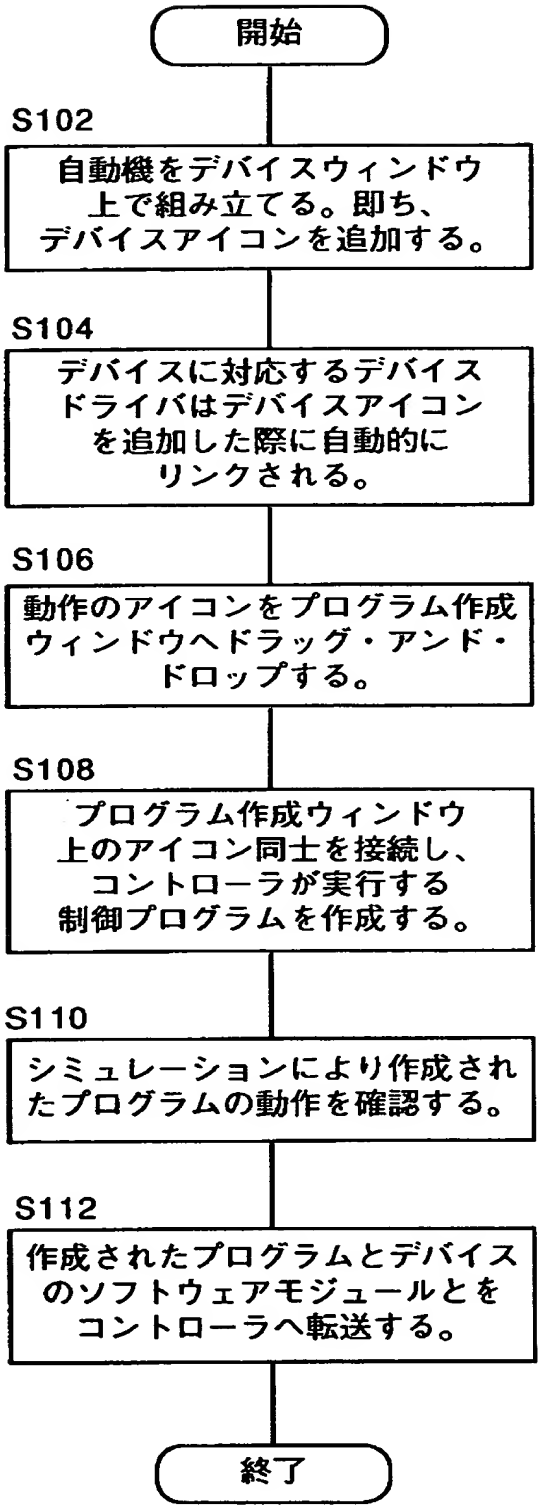
【図７】



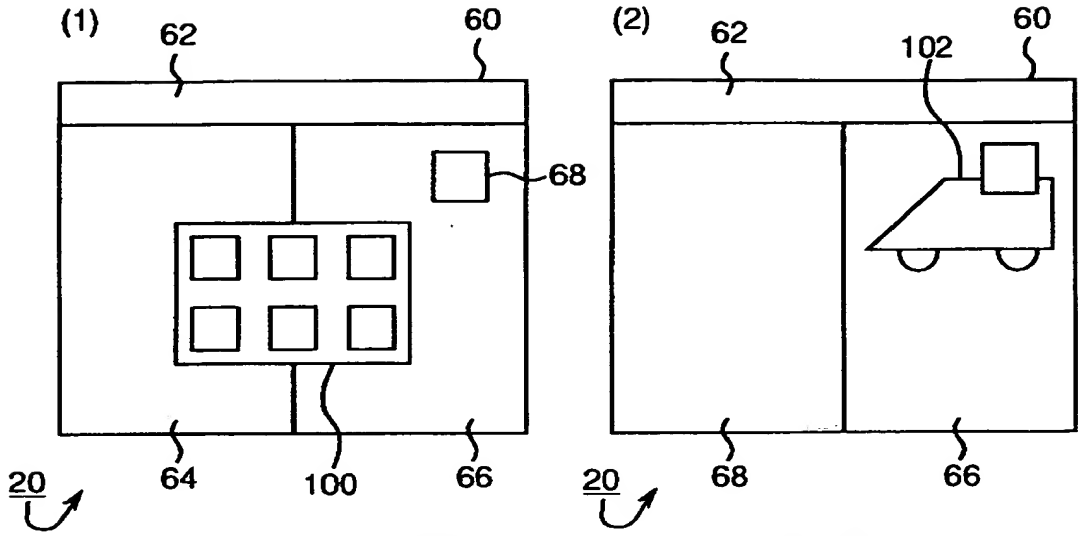
68' : デバイスアイコン
70 : 開始アイコン(制御コマンドアイコン)
72 : 矢印

68" : デバイスアイコン

【図８】

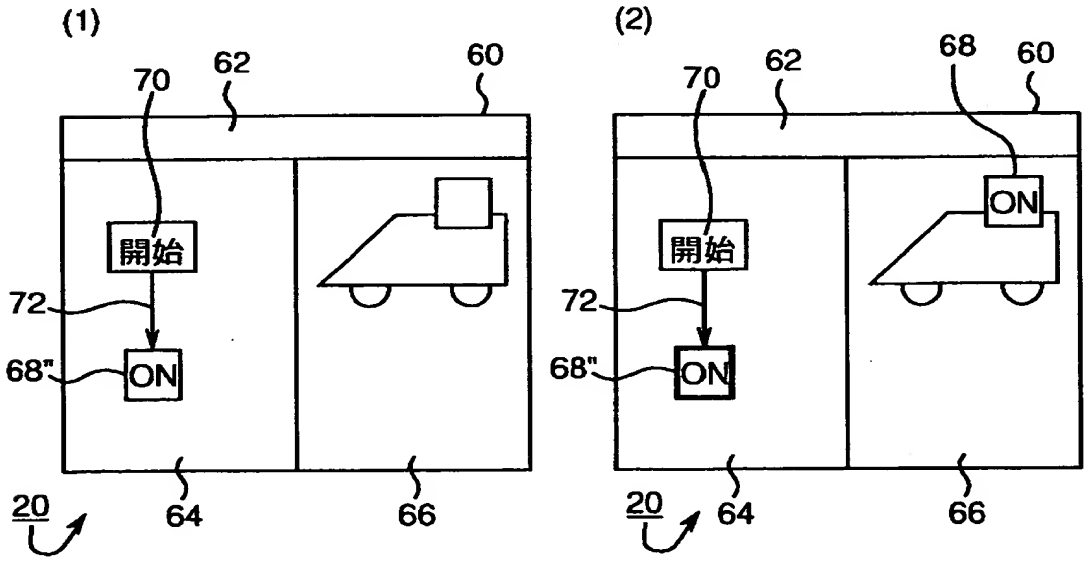

開始
S102
自動機をデバイスウィンドウ上で組み立てる。即ち、デバイスアイコンを追加する。
S104
デバイスに対応するデバイスドライバはデバイスアイコンを追加した際に自動的にリンクされる。
S106
動作のアイコンをプログラム作成ウィンドウへドラッグ・アンド・ドロップする。
S108
プログラム作成ウィンドウ上のアイコン同士を接続し、コントローラが実行する制御プログラムを作成する。
S110
シミュレーションにより作成されたプログラムの動作を確認する。
S112
作成されたプログラムとデバイスのソフトウェアモジュールとをコントローラへ転送する。
終了

【図９】

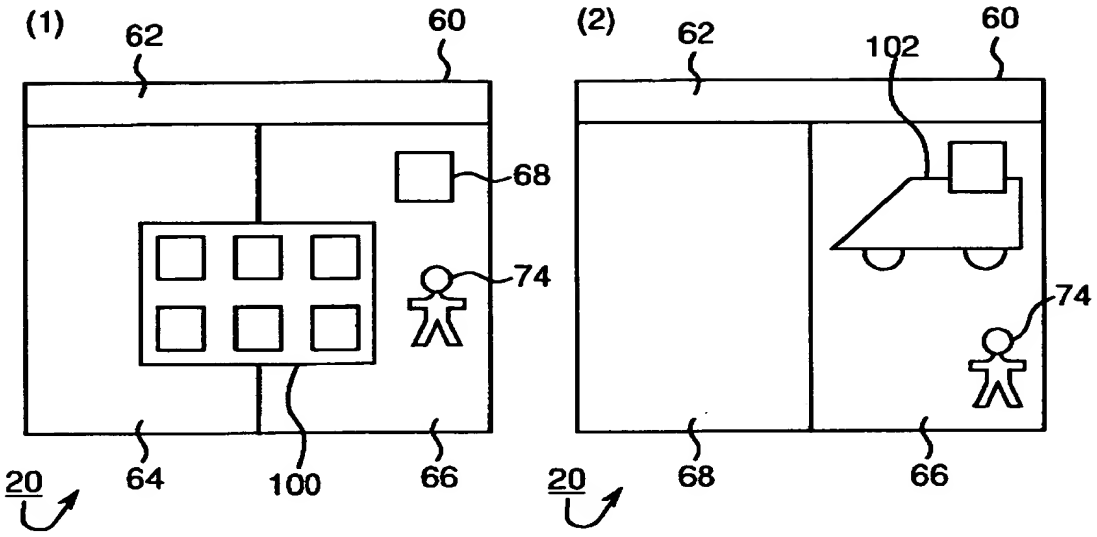


100：デバイスダイアログボックス　　102：絵

【図１０】

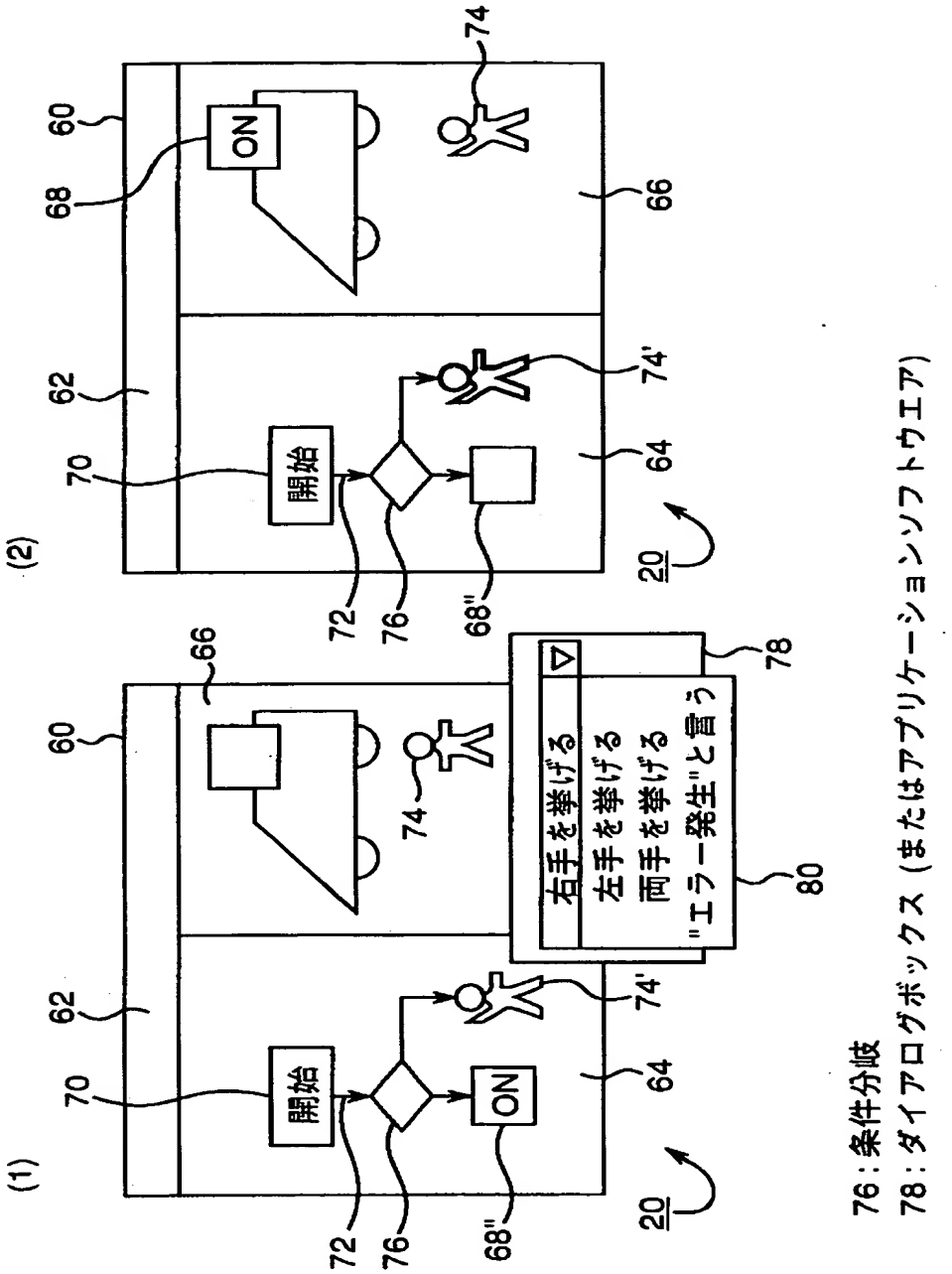


【図１１】



74：バーチャル管理人

【図１２】



76：条件分岐
78：ダイアログボックス（またはアプリケーションソフトウエア）

【書類名】　　要約書

【要約】

【課題】　自動機を構築する際に、デバイスやコントローラに係るセットアップの手間を大幅に削減するとともに、各種制御プログラムの作成を簡素化する

【解決手段】　デバイスを接続すれば必要ソフトウエアモジュールを自動的にリンクする機能を備える。デバイスアイコンや制御コマンドアイコンのダイアログボックス等、プログラミングのための各種要素を、デバイスに係る必要ソフトウエアモジュールに含める。各種制御プログラムの作成を、目的のデバイスアイコンを中心とする「デバイス指向プログラミング」により行ない、作成作業の簡素化を目指す。

【選択図】　　図１

出　願　人　履　歴　情　報

識別番号　　〔０００００６０１３〕

１．変更年月日　　１９９０年　８月２４日
［変更理由］　新規登録
住　所　東京都千代田区丸の内２丁目２番３号
氏　名　三菱電機株式会社